

BÀI 6: SỰ PHÂN TÍCH NHÓM

I. Mục tiêu:

Sau khi thực hành xong, sinh viên nắm được:

- Thuật toán K-means
- Thuật toán K-medians

II. Tóm tắt lý thuyết:

Xét tập dữ liệu D chứa n điểm dữ liệu $\overline{X_1}, \overline{X_2}, \dots, \overline{X_n}$ trong không gian d chiều. Mục tiêu xác định k biểu diễn $\overline{Y_1}, \dots, \overline{Y_k}$ cực tiểu hóa hàm mục tiêu O theo sau:

$$O = \sum_{i=1}^n \min_j \text{Dist}(\overline{X_i}, \overline{X_j})$$

1. Thuật toán K-means

Trong thuật toán k-mean, tổng bình phương các khoảng cách Euclidean của các điểm dữ liệu tới các biểu diễn gần chúng nhất được sử dụng để xác định số lượng hàm mục tiêu của việc phân cụm. Do đó, ta có

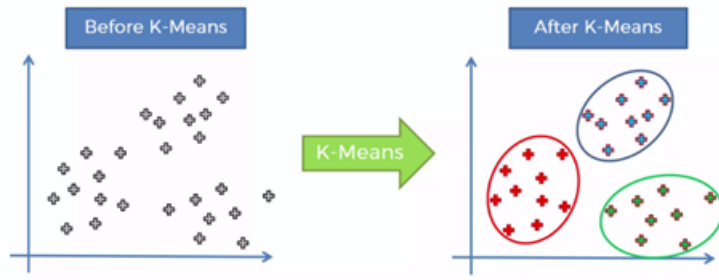
$$\text{Dist}(\overline{X_i}, \overline{X_j}) = \|\overline{X_i} - \overline{X_j}\|_2^2$$

Algorithm 1 K-Means Clustering (Lloyd's Algorithm) *Note: written for clarity, not efficiency.*

```
1: Input: Data vectors  $\{x_n\}_{n=1}^N$ , number of clusters  $K$ 
2: for  $n \leftarrow 1 \dots N$  do                                ▶ Initialize all of the responsibilities.
3:    $r_n \leftarrow [0, 0, \dots, 0]$                         ▶ Zero out the responsibilities.
4:    $k' \leftarrow \text{RandomInteger}(1, K)$                   ▶ Make one of them randomly one to initialize.
5:    $r_{nk'} = 1$ 
6: end for
7: repeat
8:   for  $k \leftarrow 1 \dots K$  do                            ▶ Loop over the clusters.
9:      $N_k \leftarrow \sum_{n=1}^N r_{nk}$                         ▶ Compute the number assigned to cluster  $k$ .
10:     $\mu_k \leftarrow \frac{1}{N_k} \sum_{n=1}^N r_{nk} x_n$           ▶ Compute the mean of the  $k$ th cluster.
11:   end for
12:   for  $n \leftarrow 1 \dots N$  do                            ▶ Loop over the data.
13:      $r_n \leftarrow [0, 0, \dots, 0]$                       ▶ Zero out the responsibilities.
14:      $k' \leftarrow \arg \min_k \|x_n - \mu_k\|^2$           ▶ Find the closest mean.
15:      $r_{nk'} = 1$ 
16:   end for
17: until none of the  $r_n$  change
18: Return assignments  $\{r_n\}_{n=1}^N$  for each datum, and cluster means  $\{\mu_k\}_{k=1}^K$ .
```

Thuật toán K-Means được tóm tắt như sau:

1. Chọn K điểm bất kỳ làm các tâm ban đầu.
2. Phân mỗi điểm dữ liệu vào cluster có center gần nó nhất.
3. Nếu việc gán dữ liệu vào từng cluster ở bước 2 không thay đổi so với vòng lặp trước nó thì ta dừng thuật toán.
4. Cập nhật tâm cho từng cluster bằng cách lấy trung bình cộng của tất cả các điểm dữ liệu đã được gán vào cluster đó sau bước 2.



5. Quay lại bước 2.

Ví dụ: sử dụng K-means để chia dữ liệu theo sau thành 2 cụm

x_1	1	2	2	3	4	5
x_2	1	1	3	2	3	5

Bước 1: Chọn ngẫu nhiên 2 tâm cụm

$$v_1 = (2, 1) \quad v_2 = (2, 3)$$

Bước 2: Tìm khoảng cách ($Dist(\bar{X}_i, \bar{X}_j) = \|\bar{X}_i - \bar{X}_j\|_2^2$) giữa các tâm cụm và mỗi điểm dữ liệu

Điểm dữ liệu	Khoảng cách từ $v_1 = (2, 1)$	Khoảng cách từ $v_2 = (2, 3)$	Gán tâm cụm
$a_1 = (1, 1)$	1	5	v_1
$a_2 = (2, 1)$	0	4	v_1
$a_3 = (2, 3)$	1	0	v_2
$a_4 = (3, 2)$	2	2	v_1
$a_5 = (4, 3)$	8	4	v_2
$a_6 = (5, 5)$	25	13	v_2

Bước 3:

Cluster 1 của v_1 : $\{a_1, a_2, a_4\}$

Cluster 2 của v_2 : $\{a_3, a_5, a_6\}$

Bước 4: Tính lại các tâm cụm

$$v_1 = \frac{1}{3}[a_1 + a_2 + a_4] = \frac{1}{3}[(1, 1) + (2, 1) + (3, 2)] = \frac{1}{3}(6, 4) = (2, 1.33)$$

$$v_2 = \frac{1}{3}[a_3 + a_5 + a_6] = \frac{1}{3}[(2, 3) + (4, 3) + (5, 5)] = \frac{1}{3}(11, 11) = (3.67, 3.67)$$

Bước 5: Lặp lại bước 2 cho tới khi tâm cụm giống nhau hoặc số phần tử cụm giống như trong bước lặp trước

Điểm dữ liệu	Khoảng cách từ $v_1 = (2, 1.33)$	Khoảng cách từ $v_2 = (3.67, 3.67)$	Gán tâm cụm
$a_1 = (1, 1)$	1.11	14.26	v_1
$a_2 = (2, 1)$	0.11	9.92	v_1
$a_3 = (2, 3)$	2.79	3.24	v_1
$a_4 = (3, 2)$	1.45	3.24	v_1
$a_5 = (4, 3)$	6.79	0.56	v_2
$a_6 = (5, 5)$	22.47	3.54	v_2

Cluster 1 của v_1 : $\{a_1, a_2, a_3, a_4\}$

Cluster 2 của v_2 : $\{a_5, a_6\}$

Tính lại các tâm cụm

$$v_1 = \frac{1}{4}[a_1 + a_2 + a_3 + a_4] = \frac{1}{4}[(1, 1) + (2, 1) + (2, 3) + (3, 2)] = \frac{1}{4}(8, 7) = (2, 1.75)$$

$$v_2 = \frac{1}{2}[a_5 + a_6] = \frac{1}{2}[(4, 3) + (5, 5)] = \frac{1}{2}(9, 8) = (4.5, 4)$$

\Rightarrow Các phần tử của cụm và tâm cụm không giống như trong bước lặp trước đó.

Điểm dữ liệu	Khoảng cách từ $v_1 = (2, 1.75)$	Khoảng cách từ $v_2 = (4.5, 4)$	Gán tâm cụm
$a_1 = (1, 1)$	1.56	21.25	v_1
$a_2 = (2, 1)$	0.56	15.25	v_1
$a_3 = (2, 3)$	1.56	7.25	v_1
$a_4 = (3, 2)$	1.06	6.25	v_1
$a_5 = (4, 3)$	5.56	1.25	v_2
$a_6 = (5, 5)$	22.47	1.25	v_2

Cluster 1 của v_1 : $\{a_1, a_2, a_3, a_4\}$

Cluster 2 của v_2 : $\{a_5, a_6\}$

\Rightarrow Các phần tử của cụm giống như trong bước lặp trước đó.

Vậy:

Cluster 1: $\{(1, 1), (2, 1), (2, 3), (3, 2)\}$

Cluster 2: $\{(4, 3), (5, 5)\}$

2. Thuật toán K-medians

Trong thuật toán K -medians, khoảng cách Manhattan được sử dụng như hàm mục tiêu. Do đó, hàm khoảng cách $Dist((\overline{X}_i, \overline{X}_j))$ được xác định như sau:

$$Dist((\overline{X}_i, \overline{X}_j)) = \|\overline{X}_i - \overline{X}_j\|_1$$

Trong trường hợp này, biểu diễn tối ưu là median của các điểm dữ liệu dọc theo mỗi chiều trong cụm C_j . Đó là bởi vì điểm dữ liệu có tổng nhỏ nhất của các khoảng cách L_1 tới một tập các điểm được phân bố trong 1 đường là median của tập hợp đó.

Trong bước cập nhật lại tâm, median được tính lại bằng việc sử dụng median trong mỗi chiều. Median được xác định như sau:

- Sắp xếp dữ liệu theo thứ tự tăng dần hay giảm dần.
- Nếu n là số lẻ thì $median =$ số hạng thứ $(\frac{n+1}{2})$. Ví dụ: xét dãy 21, 38, 46, 49, 57, 61, 64 $\Rightarrow n = 7$ là số lẻ $\Rightarrow median = (\frac{n+1}{2}) = (\frac{7+1}{2}) =$ số hạng thứ 4 = 49
- Nếu n là số chẵn thì $median = \frac{\text{số hạng thứ } (\frac{n}{2}) + \text{số hạng thứ } (\frac{n}{2} + 1))}{2}$. Ví dụ: xét dãy 2, 4, 4, 5, 7, 8 $\Rightarrow n = 6$ là số chẵn $\Rightarrow median = \frac{4+5}{2} = 4.5$

Ví dụ: sử dụng K-medians để chia dữ liệu theo sau thành 2 cụm

x_1	1	2	2	3	4	5
x_2	1	1	3	2	3	5

Bước 1: Chọn ngẫu nhiên 2 tâm cụm

$$med_1 = (2, 1) \quad med_2 = (2, 3)$$

Bước 2: Tìm khoảng cách $(Dist(\overline{X}_i, \overline{X}_j)) = \|\overline{X}_i - \overline{X}_j\|_1$ giữa các tâm cụm và mỗi điểm dữ liệu

Điểm dữ liệu	Khoảng cách từ $med_1 = (2, 1)$	Khoảng cách từ $med_2 = (2, 3)$	Gán tâm cụm
$a_1 = (1, 1)$	1	3	v_1
$a_2 = (2, 1)$	0	2	v_1
$a_3 = (2, 3)$	1	0	v_2
$a_4 = (3, 2)$	2	2	v_1
$a_5 = (4, 3)$	4	2	v_2
$a_6 = (5, 5)$	7	5	v_2

Bước 3:Cluster 1 của med_1 : $\{a_1, a_2, a_4\}$ Cluster 2 của med_2 : $\{a_3, a_5, a_6\}$ **Bước 4:** Tính lại các tâm cụm sử dụng trung vị (median)Tìm med_1 của $\{a_1, a_2, a_4\}$ - Xét dãy 1, 2, 3 $\Rightarrow median = (3 + 1)/2 =$ số hạng thứ 2 = 2.- Xét dãy 1, 1, 2 $\Rightarrow median = (3 + 1)/2 =$ số hạng thứ 2 = 1. $\Rightarrow med_1 = (2, 1)$ Tìm med_2 của $\{a_3, a_5, a_6\}$ - Xét dãy 2, 4, 5 $\Rightarrow median = (3 + 1)/2 =$ số hạng thứ 2 = 4.- Xét dãy 3, 3, 5 $\Rightarrow median = (3 + 1)/2 =$ số hạng thứ 2 = 3. $\Rightarrow med_2 = (4, 3)$ **Bước 5:** Lặp lại bước 2 cho tới khi tâm cụm giống nhau hoặc số phần tử cụm giống như trong bước lặp trước

Điểm dữ liệu	Khoảng cách từ $med_1 = (2, 1)$	Khoảng cách từ $med_2 = (4, 3)$	Gán tâm cụm
$a_1 = (1, 1)$	1	5	v_1
$a_2 = (2, 1)$	0	4	v_1
$a_3 = (2, 3)$	1	2	v_1
$a_4 = (3, 2)$	2	2	v_1
$a_5 = (4, 3)$	4	0	v_2
$a_6 = (5, 5)$	7	3	v_2

Cluster 1 của med_1 : $\{a_1, a_2, a_3, a_4\}$ Cluster 2 của med_2 : $\{a_5, a_6\}$

Tính lại các tâm cụm sử dụng trung vị (median)

Tìm med_1 của $\{a_1, a_2, a_3, a_4\}$ - Xét dãy 1, 2, 2, 3 $\Rightarrow median = \frac{2 + 2}{2} = 2$.- Xét dãy 1, 1, 3, 2 \Rightarrow sắp xếp lại dãy 1, 1, 2, 3 $\Rightarrow median = \frac{1 + 2}{2} = 1.5$ $\Rightarrow med_1 = (2, 1.5)$ Tìm med_2 của $\{a_5, a_6\}$ - Xét dãy 4, 5 $\Rightarrow median = \frac{4 + 5}{2} = 4.5$.- Xét dãy 3, 5 $\Rightarrow median = \frac{3 + 5}{2} = 4$. $\Rightarrow med_2 = (4.5, 4)$ \Rightarrow Các phần tử của cụm và tâm cụm không giống như trong bước lặp trước đó.

Điểm dữ liệu	Khoảng cách từ $med_1 = (2, 1.5)$	Khoảng cách từ $med_2 = (4.5, 4)$	Gán tâm cụm
$a_1 = (1, 1)$	1.5	6.5	v_1
$a_2 = (2, 1)$	0.5	5.5	v_1
$a_3 = (2, 3)$	1.5	3.5	v_1
$a_4 = (3, 2)$	1.5	3.5	v_1
$a_5 = (4, 3)$	3.5	1.5	v_2
$a_6 = (5, 5)$	6.5	1.5	v_2

Cluster 1 của med_1 : $\{a_1, a_2, a_3, a_4\}$

Cluster 2 của med_2 : $\{a_5, a_6\}$

\Rightarrow Các phần tử của cụm giống như trong bước lặp trước đó.

Vậy:

Cluster 1: $\{(1, 1), (2, 1), (2, 3), (3, 2)\}$

Cluster 2: $\{(4, 3), (5, 5)\}$

III. Nội dung thực hành:

1. Thuật toán K-means

- Tạo file "data.csv"

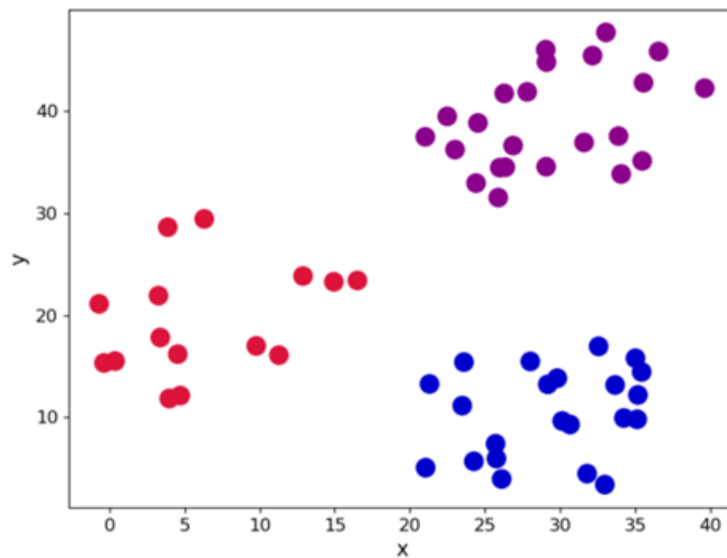
	A	B	C	D	
3	1	35.19	12.189	1	
4	2	26.288	41.718	2	
5	3	0.376	15.506	0	
6	4	26.116	3.963	1	
7	5	25.893	31.515	2	
8	6	23.606	15.402	1	
9	7	28.026	15.47	1	
10	8	26.36	34.488	2	
11	9	23.013	36.213	2	
12	10	27.819	41.867	2	
13	11	39.634	42.23	2	
14	12	35.477	35.104	2	
15	13	25.768	5.967	1	
16	14	-0.684	21.105	0	
17	15	3.387	17.81	0	
18	16	32.986	3.412	1	
19	17	34.258	9.931	1	
20	18	6.313	29.426	0	
21	19	33.899	37.535	2	
22	20	4.718	12.125	0	
23	21	21.054	5.067	1	
24	22	3.267	21.911	0	
25	23	24.537	38.822	2	
26	24	4.55	16.179	0	
27	25	25.712	7.409	1	
28	26	3.884	28.616	0	
29	27	29.081	34.539	2	
30	28	14.943	23.263	0	
31	29	32.169	45.421	2	
32	30	32.572	16.944	1	
33	31	33.673	13.163	1	
34	32	29.189	13.226	1	
35	33	25.994	34.444	2	
36	34	16.513	23.396	0	
37	35	23.492	11.142	1	
38	36	26.878	36.609	2	
39	37	31.604	36.911	2	
40	38	34.078	33.827	2	
41	39	11.286	16.082	0	
42	40	30.15	9.642	1	
43	41	36.569	45.827	2	
44	42	3.983	11.839	0	
45	43	12.891	23.832	0	
46	44	21.314	13.264	1	
47	45	29.101	44.781	2	
48	46	30.671	9.294	1	
49	47	35.139	9.803	1	
50	48	35.563	42.759	2	
51	49	35.028	15.779	1	
52	50	9.776	16.988	0	
53	51	24.268	5.693	1	
54	52	-0.36	15.319	0	
55	53	33.062	47.693	2	
56	54	21.034	37.463	2	
57	55	31.806	4.484	1	
58	56	22.493	39.466	2	
59	57	29.056	46.004	2	
60	58	29.822	13.83	1	
61	59	35.439	14.439	1	

- Đọc dữ liệu từ file

```
>>> import numpy as np
>>> import pandas as pd
>>> import matplotlib.pyplot as plt
>>> from matplotlib.colors import ListedColormap
>>> blobs = pd.read_csv('D:\\Huynh\\DataMining_Lab\\data\\tuan6\\data.csv')
>>> colnames = list(blobs.columns[1:-1])
>>> blobs.head()
  ID  x      y  cluster
0  0  24.412  32.932      2
1  1  35.190  12.189      1
2  2  26.288  41.718      2
3  3   0.376  15.506      0
4  4  26.116   3.963      1
```

- Chúng ta sẽ sử dụng cột phân cụm để hiển thị các nhóm khác nhau được biểu diễn trong tập dữ liệu

```
>>> customcmap = ListedColormap(["crimson", "mediumblue", "darkmagenta"])
>>> fig, ax = plt.subplots(figsize=(8, 6))
>>> plt.scatter(x=blobs['x'], y=blobs['y'], s=150,
                c=blobs['cluster'].astype('category'),
                cmap = customcmap)
<matplotlib.collections.PathCollection object at 0x00000245CFA11F88>
>>> ax.set_xlabel(r'x', fontsize=14)
Text(0.5, 0, 'x')
>>> ax.set_ylabel(r'y', fontsize=14)
Text(0, 0.5, 'y')
>>> plt.xticks(fontsize=12)
(array([-5., 0., 5., 10., 15., 20., 25., 30., 35., 40., 45.]), <a list of 11 Text
xticklabel objects>)
>>> plt.yticks(fontsize=12)
(array([ 0., 10., 20., 30., 40., 50.]), <a list of 6 Text yticklabel objects>)
>>> plt.show()
```



- Bước 1 và 2: Xác định k và khởi tạo các tâm

```
>>> def initiate_centroids(k, dset):
    """
    Select k data points as centroids
    k: number of centroids
    dset: pandas dataframe
    """
    centroids = dset.sample(k)
    return centroids

>>> np.random.seed(42)
>>> k=3
>>> df = blobs[['x', 'y']]
>>> centroids = initiate_centroids(k, df)
>>> centroids
      x      y
0  24.412  32.932
5  25.893  31.515
36 26.878  36.609
```

- Bước 3: tính khoảng cách

```
>>> def rsserr(a,b):
    '''
    Calculate the root of sum of squared errors.
    a and b are numpy arrays
    '''
    return np.square(np.sum((a-b)**2))
```

```
>>> for i, centroid in enumerate(range(centroids.shape[0])):
    err = rsserr(centroids.iloc[centroid,:], df.iloc[36,:])
    print('Error for centroid {0}: {1:.2f}'.format(i, err))

Error for centroid 0: 384.22
Error for centroid 1: 724.64
Error for centroid 2: 0.00
```

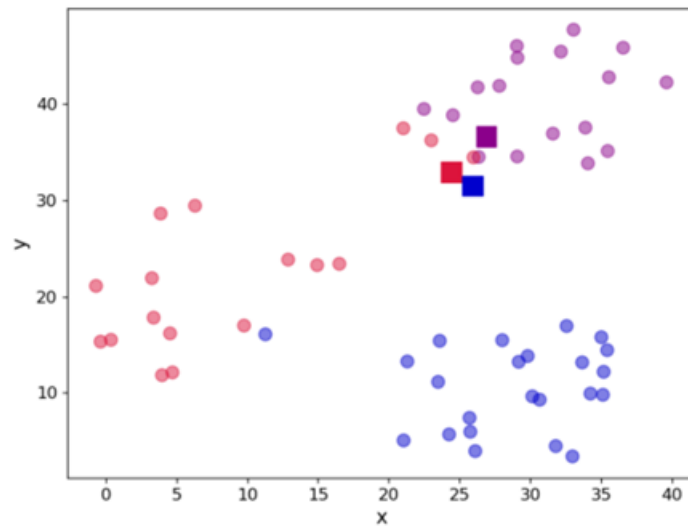
- Bước 4: gán giá trị các tâm

```
>>> def centroid_assignment(dset, centroids):
    '''
    Given a dataframe 'dset' and a set of 'centroids', we assign each
    data point in 'dset' to a centroid.
    - dset - pandas dataframe with observations
    - centroids - pandas dataframe with centroids
    '''
    k = centroids.shape[0]
    n = dset.shape[0]
    assignation = []
    assign_errors = []
    for obs in range(n):
        # Estimate error
        all_errors = np.array([])
        for centroid in range(k):
            err = rsserr(centroids.iloc[centroid, :], dset.iloc[obs,:])
            all_errors = np.append(all_errors, err)
        # Get the nearest centroid and the error
        nearest_centroid = np.where(all_errors==np.min(all_errors))[0].tolist()[0]
        nearest_centroid_error = np.min(all_errors)
        # Add values to corresponding lists
        assignation.append(nearest_centroid)
        assign_errors.append(nearest_centroid_error)
    return assignation, assign_errors
```

- Thêm cột gán tâm và sai số phát sinh để cập nhật biểu đồ phân tán biểu diễn các trọng tâm

```
>>> df['centroid'], df['error'] = centroid_assignment(df, centroids)
>>> df.head()
   x      y  centroid      error
0 24.412 32.932        0  0.000000
1 35.190 12.189        1 211534.211314
2 26.288 41.718        2   699.601495
3  0.376 15.506        0 776856.744109
4 26.116  3.963        1 576327.599678
```

```
>>> fig, ax = plt.subplots(figsize=(8, 6))
>>> plt.scatter(df.iloc[:,0], df.iloc[:,1], marker = 'o',
               c=df['centroid'].astype('category'),
               cmap = customcmap, s=80, alpha=0.5)
<matplotlib.collections.PathCollection object at 0x00000245D05DA648>
>>> plt.scatter(centroids.iloc[:,0], centroids.iloc[:,1],
               marker = 's', s=200, c=[0, 1, 2],
               cmap = customcmap)
<matplotlib.collections.PathCollection object at 0x00000245D05DA488>
>>> ax.set_xlabel(r'x', fontsize=14)
Text(0.5, 0, 'x')
>>> ax.set_ylabel(r'y', fontsize=14)
Text(0, 0.5, 'y')
>>> plt.xticks(fontsize=12)
(array([-5.,  0.,  5., 10., 15., 20., 25., 30., 35., 40., 45.]), <a list of 11 Text xticklabel objects>)
>>> plt.yticks(fontsize=12)
(array([ 0., 10., 20., 30., 40., 50.]), <a list of 6 Text yticklabel objects>)
>>> plt.show()
```

- Tổng các sai số

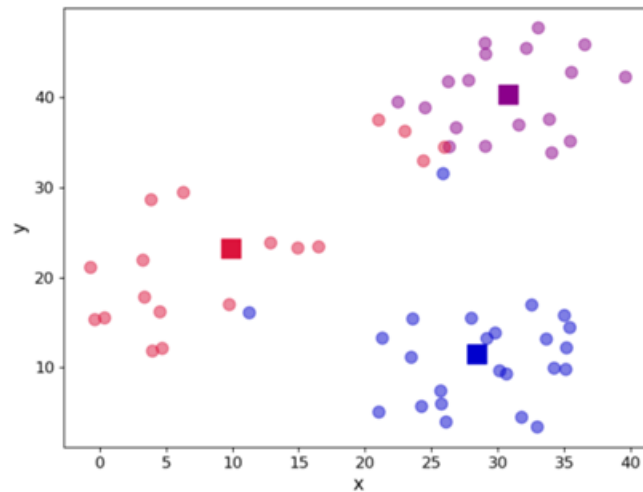
```
>>> print("The total error is {0:.2f}".format(df['error'].sum()))
The total error is 11927659.01
```

- Bước 5: cập nhật vị trí của k tâm bằng việc tính giá trị trung bình của các quan sát được gán cho mỗi tâm

```
>>> centroids = df.groupby('centroid').agg('mean').loc[:, colnames].reset_index(drop = True)
>>> centroids
   x      y
0  9.889444 23.242611
1 28.435750 11.546250
2 30.759333 40.311167
```

- Xem lại biểu đồ phân tán với vị trí của k tâm đã được cập nhật

```
>>> fig, ax = plt.subplots(figsize=(8, 6))
>>> plt.scatter(df.iloc[:,0], df.iloc[:,1], marker = 'o',
               c=df['centroid'].astype('category'),
               cmap = customcmap, s=80, alpha=0.5)
<matplotlib.collections.PathCollection object at 0x00000245CFA4E748>
>>> plt.scatter(centroids.iloc[:,0], centroids.iloc[:,1],
               marker = 's', s=200,
               c=[0, 1, 2], cmap = customcmap)
<matplotlib.collections.PathCollection object at 0x00000245D05DA308>
>>> ax.set_xlabel(r'x', fontsize=14)
Text(0.5, 0, 'x')
>>> ax.set_ylabel(r'y', fontsize=14)
Text(0, 0.5, 'y')
>>> plt.xticks(fontsize=12)
(array([-5., 0., 5., 10., 15., 20., 25., 30., 35., 40., 45.]), <a list of 11 Text xticklabel objects>)
>>> plt.yticks(fontsize=12)
(array([ 0., 10., 20., 30., 40., 50.]), <a list of 6 Text yticklabel objects>)
>>> plt.show()
```



- Bước 6: lặp lại bước 3-5

```
>>> def kmeans(dset, k=2, tol=1e-4):
    """
    K-means implementation for a
    `dset`: DataFrame with observations
    `k`: number of clusters, default k=2
    `tol`: tolerance=1E-4
    """
    # Let us work in a copy, so we don't mess the original
    working_dset = dset.copy()
    # We define some variables to hold the error, the
    # stopping signal and a counter for the iterations
    err = []
    goahead = True
    j = 0

    # Step 2: Initiate clusters by defining centroids
    centroids = initiate_centroids(k, dset)

    while goahead:
        # Step 3 and 4 - Assign centroids and calculate error
        working_dset['centroid'], j_err = centroid_assignment(working_dset, centroids)
        err.append(sum(j_err))

        # Step 5 - Update centroid position
        centroids = working_dset.groupby('centroid').agg('mean').reset_index(drop = True)

        # Step 6 - Restart the iteration
        if j>0:
            # Is the error less than a tolerance (1E-4)
            if err[j-1]-err[j]<=tol:
                goahead = False
            j+=1

    working_dset['centroid'], j_err = centroid_assignment(working_dset, centroids)
    centroids = working_dset.groupby('centroid').agg('mean').reset_index(drop = True)
    return working_dset['centroid'], j_err, centroids
```

- Thực thi hàm trên

```
>>> np.random.seed(42)
>>> df['centroid'], df['error'], centroids = kmeans(df[['x','y']], 3)
>>> df.head()
```

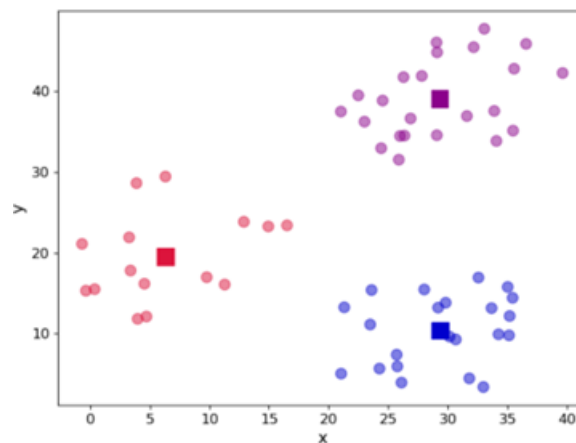
	x	y	centroid	error
0	24.412	32.932	2	3767.568743
1	35.190	12.189	1	1399.889001
2	26.288	41.718	2	262.961097
3	0.376	15.506	0	2683.086425
4	26.116	3.963	1	2723.650198

- Vị trí của các tâm cuối cùng

```
>>> centroids
      x      y
0  6.322867 19.559800
1 29.330864 10.432409
2 29.304957 39.050783
```

- Xem lại biểu đồ phân tán

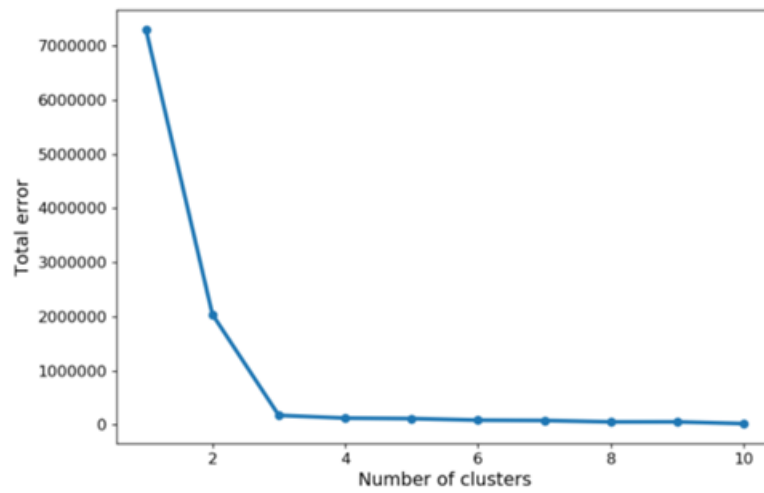
```
>>> fig, ax = plt.subplots(figsize=(8, 6))
>>> plt.scatter(df.iloc[:,0], df.iloc[:,1], marker = 'o',
               c=df['centroid'].astype('category'),
               cmap = customcmap, s=80, alpha=0.5)
<matplotlib.collections.PathCollection object at 0x00000245CF2F4B48>
>>> plt.scatter(centroids.iloc[:,0], centroids.iloc[:,1],
               marker = 's', s=200, c=[0, 1, 2],
               cmap = customcmap)
<matplotlib.collections.PathCollection object at 0x00000245CF2FD608>
>>> ax.set_xlabel(r'x', fontsize=14)
Text(0.5, 0, 'x')
>>> ax.set_ylabel(r'y', fontsize=14)
Text(0, 0.5, 'y')
>>> plt.xticks(fontsize=12)
(array([-5., 0., 5., 10., 15., 20., 25., 30., 35., 40., 45.]), <a list of 11 Text xticklabel objects>)
>>> plt.yticks(fontsize=12)
(array([ 0., 10., 20., 30., 40., 50.]), <a list of 6 Text yticklabel objects>)
>>> plt.show()
```



- Sử dụng “elbow” để chỉ ra số cụm tối ưu

```
>>> err_total = []
>>> n = 10
>>> df_elbow = blobs[['x', 'y']]
>>> for i in range(n):
    _, my_errs, _ = kmeans(df_elbow, i+1)
    err_total.append(sum(my_errs))

>>> fig, ax = plt.subplots(figsize=(8, 6))
>>> plt.plot(range(1,n+1), err_total, linewidth=3, marker='o')
<matplotlib.lines.Line2D object at 0x00000245CF53CE88>
>>> ax.set_xlabel(r'Number of clusters', fontsize=14)
Text(0.5, 0, 'Number of clusters')
>>> ax.set_ylabel(r'Total error', fontsize=14)
Text(0, 0.5, 'Total error')
>>> plt.xticks(fontsize=12)
(array([ 0., 2., 4., 6., 8., 10., 12.]), <a list of 7 Text xticklabel objects>)
>>> plt.yticks(fontsize=12)
(array([-1000000., 0., 1000000., 2000000., 3000000., 4000000.,
        5000000., 6000000., 7000000., 8000000.]), <a list of 10 Text yticklabel objects>)
>>> plt.show()
```



2. Thuật toán K-medians

- Làm tương tự với thuật toán ở mục 1.
- Trình bày tóm tắt lại phần code trong file báo cáo.